NUWC-NPT Technical Report 10,472
30 June 1995

Applying Random Search Algorithms to Target Motion Analysis

D. J. Ferkinhoff
S. E. Hammel
K. F. Gong
Combat Systems Department

Naval Undersea Warfare Center Division
Newport, Rhode Island

PREFACE

This report was prepared under Project No. 33CWF55, principal investigator A.H. Silva (Code 2211). The sponsoring activity is the Office of Naval Research (James Fein, ONR-333).

The technical reviewer for this report was R. Kargher (Code 2211).

Reviewed and Approved: 30 June 1995

P. A. La Brecque
Head, Combat Systems Department

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
30 June 1995

3. REPORT TYPE AND DATES COVERED

4. TITLE AND SUBTITLE

Applying Random Search Algorithms to Target Motion Analysis

5. FUNDING NUMBERS

6. AUTHOR(S)

D. J. Ferkinhoff
S. E. Hammel
K. F. Gong

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Undersea Warfare Center Division
1176 Howell Street
Newport, RI 02841-1708

8. PERFORMING ORGANIZATION REPORT NUMBER

TR 10,472

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This report addresses the use of random search algorithms for estimating contact state parameters. Specifically, simulated annealing and genetic algorithms are developed and their performance is examined. In addition, a method is presented for using these algorithms as either stand-alone target state estimation techniques, or as methods for initializing gradient or grid-based estimation techniques in a hybrid system. Performance of these algorithms is examined via Monte Carlo simulation using surface ship active data.

14. SUBJECT TERMS

Target Motion Analysis	Contact Tracking
Genetic Algorithms	Simulated Annealing

15. NUMBER OF PAGES
34

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

TABLE OF CONTENTS

Section Page

LIST OF ILLUSTRATIONS

Figure Page

LIST OF TABLES

Table Page

APPLYING RANDOM SEARCH ALGORITHMS TO TARGET MOTION ANALYSIS

I. INTRODUCTION

Contact tracking involves processing data from various sensors to provide an estimate of a contact's position and velocity, or state. Reliable and unique contact state estimates can be obtained under favorable noise, geometric and environmental conditions, or highly observable conditions. However, most practical situations do not fulfill these conditions, which together with the inherent uncertainty in selecting appropriate mathematical models, can cause instability in the estimation process. In addition, the relationship between the contact state and the observed measurements is nonlinear; therefore, any linearization procedures applied can introduce additional estimation errors. Under these conditions, alternative algorithms that provide efficient and reliable estimates are needed.

Various gradient-based estimation techniques, such as extended Kalman filters or maximum likelihood estimators, are available to provide tracking estimates by searching for the peak of the target state density function (references 1 - 3). However, these techniques employ a search procedure based on the local gradient of the density function, which can lead to convergence to local maxima. Another potential problem associated with these algorithms is that they can diverge when the problem becomes ill-conditioned, such as when the measurements are very noisy or the data are sparse and intermittent. These conditions are especially prevalent when tracking with active or passive data in a shallow-water environment.

Because of their processing stability, grid-based techniques have recently been applied to the target state estimation problem (reference 4). Unlike their gradient-based counterparts, these techniques estimate the unknown contact parameters by direct reconstruction of the state density function. In this process, a grid of predetermined size and resolution is typically used, and the value of the density function is computed at all grid points. In principle, this computationally expensive technique can provide the desired efficacy; however, it lacks efficiency. In addition, the grid must be properly placed, and the resolution and size must be appropriately selected to properly represent the contact state density.

This report focuses on the application of random search techniques to contact tracking. Specifically, a simulated annealing algorithm (SA) and a genetic algorithm (GA) are developed as search mechanisms for the contact state estimation problem. For this investigation, the applicability of SA and GA as initialization schemes or as stand-alone tracking algorithms is examined. As will be shown, these random search algorithms do not use gradients and can be more efficient than grid-based algorithms. Thus, they are promising candidates for solving the contact tracking problem under poor observability and/or multimodal conditions.

This report is structured as follows. First, a contact tracking system concept is proposed. Next, the state and measurement equations are described, along with a traditional gradient-based maximum likelihood estimator (MLE). SA and GA tracking techniques are then developed. The efficiency and efficacy of these techniques are illustrated by a simulation example of surface ship tracking with sparse and intermittent active data. Finally, the findings and observations are summarized.

II. TRACKING SYSTEM CONCEPT

Measurements generated from various sensors within a sonar suite contain the information necessary to determine contact state parameters. Due to the noisy ocean environment, the measurements are first preprocessed, which includes data editing, data association, pre-whitening and data segmentation (references 5 and 6). The data are next processed by data association and tracking algorithms to provide estimates of contact state parameters. The resulting estimates are then evaluated for accuracy and statistical consistency. In principle, a combination of algorithms can be applied to provide the desired efficiency and efficacy. For example, appropriate algorithms can be selected based on data type, trends observed in the data, system observability, or environmental conditions. Alternatively, a selected algorithm can be used to provide an initial estimate of target state parameters to another algorithm or algorithms for refinement. This concept is illustrated in figure 1.

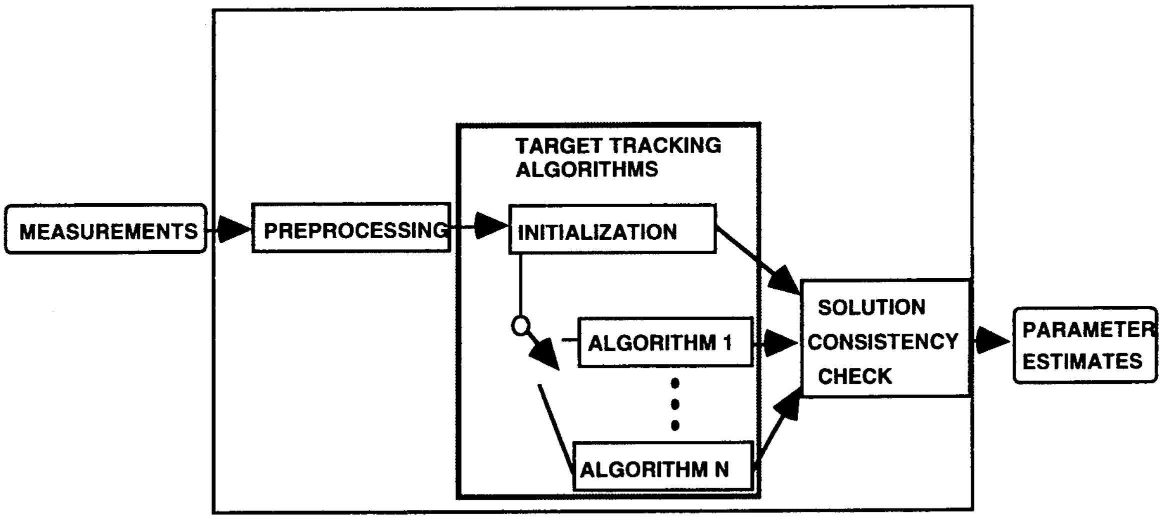


Figure 1. Contact State Estimation System

III. SYSTEM EQUATIONS AND MAXIMUM LIKELIHOOD SOLUTION

A typical state estimation algorithm employs models of platform kinematics, the environment, and sensors. Without loss of generality, the contact is assumed to be of constant velocity, while own ship is free to maneuver. Further, straight-line signal propagation is assumed.

Let the contact state vector x_T be defined as

$$X_T = \left[R_{XT}(t_o) R_{YT}(t_o) V_{XT} V_{YT}\right]^T, \tag{1a}$$

where $R_{XT}(t_o)$ and $R_{YT}(t_o)$ are the Cartesian position components at time t_o, and V_{XT} and V_{YT} are the corresponding velocity components.

The observer state is similarly defined as

$$X_O = \left[R_{XO}(t_o), R_{YO}(t_o), V_{XO}, V_{YO}\right]^T. \tag{1b}$$

The contact state relative to the observer is defined as

$$X(t_o) = X_T - X_{OS} = \left[R_X(t_o), R_Y(t_o), V_X, V_Y\right]^T, \tag{1c}$$

where $R_X(t_o)$ and $R_Y(t_o)$ are the relative Cartesian position components at time t_o, and V_X and V_Y are the relative velocity components. Let t_i be the i^{th} sampling time. The state dynamic equations are governed by

$$X(t_{i+1}) = \Phi(t_{i+1}, t_i) X(t_i) + u(t_i), \tag{2a}$$

where $\Phi(t_{i+1}, t_i)$ is the state transition matrix defined as

$$\Phi(t_{i+1}, t_i) = \begin{bmatrix} I_{2x2} & (t_{i+1} - t_i) I_{2x2} \\ 0 & I_{2x2} \end{bmatrix}, \tag{2b}$$

with I_{2x2} being a two-dimensional identity matrix, and $u(t_i)$ is a vector relating to own ship acceleration at time t_i. The measurement vector Z is

$$Z = H(X) + \eta, \tag{3a}$$

where $H(X)$ is a nonlinear function relating Z to the state X; and for m measurements of bearing, B_i and range, R_i,

$$H(X) = \begin{bmatrix} \beta_o \\ \vdots \\ \beta_m \\ \\ R_o \\ \vdots \\ R_m \end{bmatrix} = \begin{bmatrix} a\tan\left(\frac{R_x(t_o)}{R_Y(t_o)}\right) \\ \vdots \\ a\tan\left(\frac{R_x(t_m)}{R_Y(t_m)}\right) \\ \\ \sqrt{R_X(t_o)^2 + R_y(t_o)^2} \\ \vdots \\ \sqrt{R_X(t_m)^2 + R_y(t_m)^2} \end{bmatrix}, \tag{4}$$

and η is the white, Gaussian noise vector defined as

$$\eta = \left[\eta_{\beta o}\eta_{\beta 1}\cdots\eta_{\beta m}\eta_{Ro}\eta_{R1}\cdots\eta_{Rm}\right]^T, \tag{5a}$$

with mean and covariance

$$E[\eta] = 0, \tag{5b}$$

$$E[\eta\eta^T] = W = \begin{bmatrix} \sigma^2_{\beta o} & & 0 & \\ \ddots & \sigma^2_{\beta m} & & \\ & & \sigma^2_{Ro} & \\ & 0 & \ddots & \sigma^2_{Rm} \end{bmatrix}. \tag{5c}$$

As shown in reference 7, determining the maximum likelihood estimate is equivalent to finding the $\hat{X}$ that minimizes the cost function, $\|Z - H(X)\|^2 w^{-1}$; i.e.,

$$\hat{X}_{MLE} = \min_{\hat{x}}\left\{\|Z - H(X)\|^2_{w^{-1}}\right\} \tag{6}$$

Performing the above operation yields

$$\hat{X} = \left[\Phi^T J^T W^{-1} J\Phi\right]^{-1} \Phi^T J^T W^{-1} Z, \tag{7}$$

where

$$J = \left.\frac{\partial H(X)}{\partial X}\right|_{x = \hat{x}}, \tag{8}$$

is the Jacobian.

The term $\left[\Phi^T J^T W^{-1} J\Phi\right]$ in equation (7) is the Fisher Information Matrix (FIM), which must be nonsingular for $\hat{X}$ to be uniquely determined from the data (references 8 and 9). Because this is a gradient-based technique, the cost function and its derivative must be continuous. Inherent to the problem formulation are assumed system models. However, in many situations the models may not be known exactly. Traditional methods of solving the nonlinear tracking problem are sensitive to noise and geometric conditions, as well as modeling, linearization, and initialization errors. These sources of error can cause problems by injecting errors in the computation of J and, thus, the FIM. As such, these methods may be prone to ill-conditioning and instability (references 10 - 12).

To enhance the estimation process, random search algorithms can be applied. Because they are able to search the state space without computing the gradient or inverting the FIM, they provide the potential for alleviating some of the instabilities associated with gradient-based techniques. In addition, they are not affected by discontinuities in the cost surface and can potentially provide the desired efficacy and efficiency. To examine their potential, the derivation of SA and GA for contact tracking applications is presented in the following section.

IV. SIMULATED ANNEALING

Simulated annealing takes its name from a mechanical process known as annealing. In this process, a metal (or combination of metals) is first heated and then cooled at a particular rate. The cooling rate is controlled by a temperature schedule appropriate to allow the metallic crystals to form in the desired manner. This process provides the desired characteristics of the final product by minimizing the internal stresses or energy (references 13 and 14). Adapting SA for contact tracking mimics this phenomenon. Specifically, finding a state that minimizes the cost function, or equivalently maximizes the contact state density, via the annealing process, is analogous to arranging the atomic state of the metal such that the internal energy of the metal is minimized.

For the tracking problem, the estimation process via SA starts with a random guess $\hat{X}_o$; however, some deterministic knowledge can be employed. The desired solution is obtained iteratively where the estimate at the k+1th iteration is computed by adding a small random perturbation, ΔX, to the kth estimate; i.e.,

$$\hat{X}_{k+1} = \hat{X}_k + \Delta X, \tag{9a}$$

where

$$\Delta X = \begin{bmatrix} N\left(0, \Delta^2{}_{Rx}\right) \\ N\left(0, \Delta^2{}_{Ry}\right) \\ N\left(0, \Delta^2{}_{Vx}\right) \\ N\left(0, \Delta^2{}_{Vy}\right) \end{bmatrix}, \tag{9b}$$

with $N(0,\Delta^2{}_{(.)})$being a white, Gaussian-distributed, random variable with zero mean and variances $\Delta^2{}_{(.)}$ for the (.) parameter. Note that no gradient information is used in computing ΔX, and that this technique does not require the cost function to be continuous. The change in the cost, or energy, is computed via

$$\Delta E = E\left(\hat{X}_{k+1}\right) - E\left(\hat{X}_k\right), \tag{10}$$

where

$$E\left(\hat{X}_k\right) = \left\| Z - H\left(\hat{X}_k\right) \right\|^2_{W^{-1}} \ . \tag{11}$$

If ΔE is less than zero, indicating a search in the direction of minimum energy, $\hat{X}_{k+1}$ is accepted; otherwise, the probability of accepting $\hat{X}_{k+1}$ is computed via (reference 13):

$$\tau\left(\hat{X}_{k+1}\right)=\exp\left(\frac{-\Delta E}{T_j}\right) , \tag{12}$$

where T_j is the temperature at the j^{th} temperature iteration. To give the algorithm an ability to settle somewhat at the j^{th} temperature, the temperature was only updated every I_T perturbations of the state estimate, where I_T was chosen to be a small value (<10). For the problem at hand, the temperature schedule was selected as

$$T_j=\alpha^{j}T_o \ , \tag{13}$$

where T_o is the initial temperature and α is a constant less than unity. A minimum value for T_j is also specified. Equation (13) is determined empirically, as standards for optimal temperature selection for this problem do not exist.

The decision to accept $\hat{X}_{k+1}$ as the new estimate when ΔE is positive is made by comparing $\tau\left(\hat{X}_{k+1}\right)$ to a random number generated from a uniform distribution between zero and one; i.e., accept $\hat{X}_{k+1}$ if

$$\tau\left(\hat{X}_{k+1}\right)>U[0,1] \ ; \tag{14a}$$

otherwise,

$$\hat{X}_{k+1}=\hat{X}_k \ . \tag{14b}$$

The entire procedure is iterated until a pre-specified convergence criterion is satisfied. Here, the convergence criterion employed includes the use of a minimum cost, a maximum number of iterations, and a minimum change in the contact state estimate. The estimation process employing the simulated annealing search procedure is shown in figure 2. If the algorithm was determined to have prematurely converged, it was reinitialized at a different initial estimate, where the maximum number of re-annealings was specified. The SA target tracking algorithm is summarized in table 1.

The design of the algorithm provides the following properties. With a sufficiently high temperature, any change in the contact state estimate, regardless of change in cost, will have a high probability of being accepted. Here, it is assumed that the change in contact state is in fact moving toward the minimum cost, even though the cost may have increased for this iteration. As the temperature decreases, the probability of accepting the changes in the estimate that cause large increases in cost also decreases. This mechanism allows the estimate to move out of local minima, while maintaining the search toward the minimum cost. For example, when the temperature is infinite all changes in the estimate are accepted; but, for zero temperature, only changes that decrease the cost are accepted. Intermediate temperatures change the probability of accepting those changes in the state estimate with increased cost where the probability is a function of the change in cost and the temperature.

Table 1. Summary of Simulated Annealing Algorithm

1: Make a random guess for the initial target state.	2: Compute the cost (energy) of the target state estimate.
3: Randomly generate a small change in the target state estimate.	4: Compute the cost of the new target state estimate.
5: Check new cost against stopping criteria; if stopping criteria are met, stop; otherwise, go to step 6.	6: If enough iterations have been performed since the last temperature update, change the temperature; otherwise, increment the temperature counter.
7: Determine if change in energy is negative; if so, accept change in target state and go to step 10; if not, go to step 8.	8: Compute the probability of accepting the new target state.
9: Compare the probability to a random threshold; if the probability exceeds the threshold, go to step 10; otherwise, go to step 3.	10: Update the target state estimate. Go to step 3.

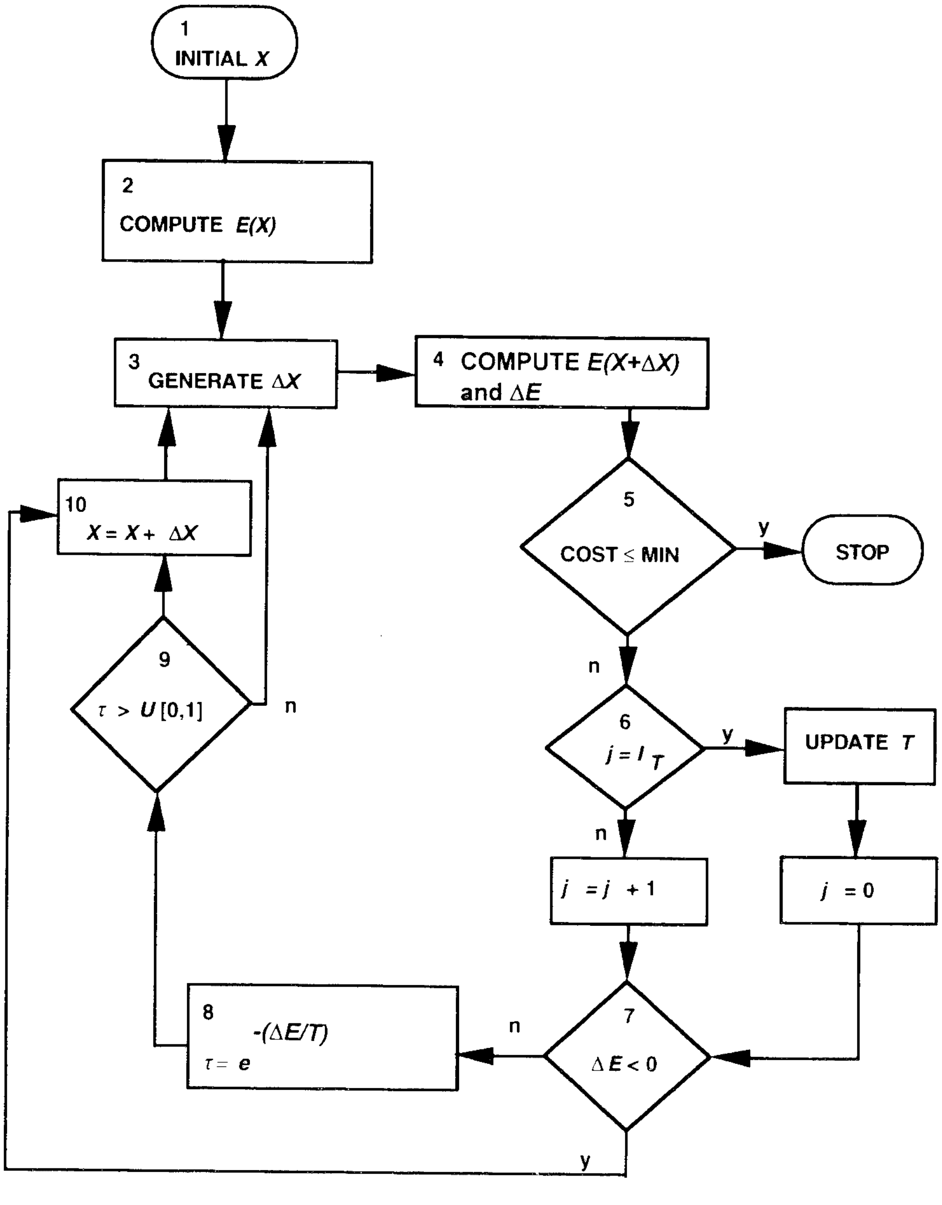


Figure 2. Flowchart of Simulated Annealing Algorithm

V. GENETIC ALGORITHMS

The genetic algorithm (GA) takes its name from the study of genetics in biology where the rule in nature is survival of the fittest. In this process, individuals in the population with the best genes suited for the local environment have a better chance of surviving to produce offspring, thus passing their genes to the next generation. Here, a global environment can have several local environments, each having an associated set of appropriate genes. For instance, a given geographic location can have a forested area that supports browsers, while a nearby plain can support grazers. The phenomenon in which different gene sequences are more appropriate for the different local environments is known as niche sharing. The genetic method of propagating genes to subsequent populations is through the use of three probabilistic mechanisms: parent selection (which allows the best individuals from the current population to have a higher probability of being selected), crossover or mating (which forms new genes by combining sequences from the parents and passes the new genes along to children), and mutation (which helps to prevent loss of genetic information).

Adapting the GA for contact tracking mimics the survival of the fittest rule by defining a binary coding of the contact state variables and operating on the bits in the same manner as genes in biology. For the state estimation problem, the process of determining which genes are best suited for the local environments is equivalent to finding the maxima in a multimodal density function (reference 15). Thus, the problem becomes one of finding the bit sequences that, after converting to real numbers, determine the locations of the various maxima in the contact state density function or equivalently, the minima in the cost function. The parent selection, crossover, and mutation mechanisms, as applied to the tracking problem, are described in the following sections.

PARENT SELECTION

Parent selection is a probabilistic method for selecting the best-fit individuals (or samples) for mating from a population of size P. Each sample of the population has an associated fitness or performance value, $perf\left(\hat{X}_i\right)$. Without loss of generality, let

$$perf\left(\hat{X}_i\right) = \frac{1}{\|Z - H(X)\|^2 w^{-1}}\Bigg| X = \hat{X}_i \quad i = 1,2,\ldots P. \tag{15}$$

In the parent selection process, stochastic errors in sampling caused by small P can lead to excessive self replication by high performance samples. This can result in a clustering of these samples about one maximum of the state density function, or local environment (reference 15). For the tracking problem, many situations warrant finding all maxima in the state density function. Thus, a mechanism similar to the one which produces niche sharing must be used to distribute samples among other peaks in the state density function, and care must be taken to select a large enough population size to facilitate niche sharing.

For the application considered here, niche sharing is facilitated by scaling the performance values relative to the distance among all samples in the population. Specifically, let the sum of the Euclidean distances from the k^{th} sample to all others be defined as

$$D\left(\hat{X}_k\right)=\sum_{i=0}^{P}\left\|\hat{X}_k-\hat{X}_i\right\|^2_{\Omega_k} \quad , \tag{16}$$

where Ω_k is a weighting vector,

$$\Omega_k=\begin{bmatrix}\omega_{Rxk}\\ \omega_{Ryk}\\ \omega_{Vxk}\\ \omega_{Vyk}\end{bmatrix} . \tag{17}$$

For $d \in \left\{R_{Xk}, R_{Yk}, V_{Xk}, V_{Yk}\right\}$, with corresponding maximum $d_{\max}$ and minimum $d_{\min}$, the components for the weighting vector in the k^{th} sample are defined as

$$\omega_{dk}\left\{1.0+1.9\left[\frac{\hat{d}-\dfrac{d_{max}-d_{min}}{2.0}}{d_{max}-d_{min}}\right]\right\}^{-1} . \tag{18}$$

The performance function can now be scaled by $D\left(\hat{X}_k\right)$ as

$$pf1\left(\hat{X}_k\right)=D\left(\hat{X}_k\right)perf\left(\hat{X}_k\right) \quad , \tag{19a}$$

which is subsequently normalized to reflect the probability of selection as

$$pf\left(\hat{X}_k\right)=\frac{pf1\left(\hat{X}_k\right)}{\sum_{i=1}^{P}pf1\left(\hat{X}_k\right)} \qquad k=1,2,\ldots P \quad . \tag{19b}$$

Parent selection can now be performed P times, i.e., select X_L if

$$\min_{L}\left[\sum_{i=1}^{L}pf\left(\hat{X}_i\right)\right]>U[0,1] \quad , \tag{20}$$

where $U[0,1]$ is a random number obtained from a uniform distribution between zero and one. The parent selection process is illustrated in figure 3.

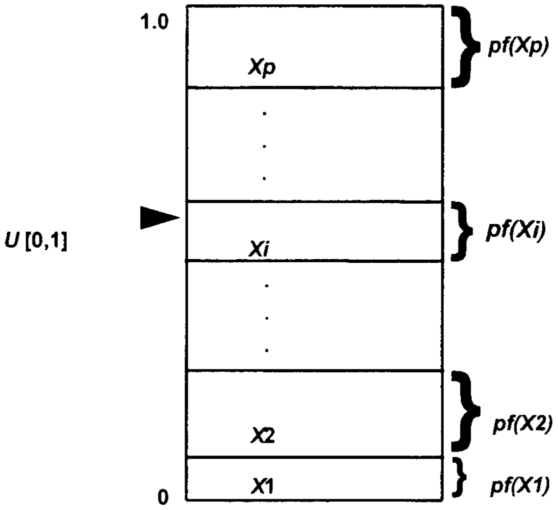


Figure 3. Illustration of Parent Selection

CROSSOVER

Once two parents are selected, crossover is performed based on a pre-specified probability. The combination of parent selection and crossover is the major search mechanism of the algorithm; therefore, the probability of crossover is typically greater than 90 percent. If crossover is not performed, the parents are copied identically to the child population. When crossover is performed, a crossover site is first randomly chosen in the bit string, where the same crossover site is used for both parents to preserve the length of the bit strings. The two sub-strings located after the crossover site for the two parents are exchanged to create two new strings, or children.

Multiple crossing sites can also be used, in which case every other sub-string is exchanged. Note that using more crossing sites will scramble the longer gene sequences, while using fewer crossing sites will result in fewer combinations of genes. Thus, the desired stability of the gene sequences should be taken into account when determining the number of crossing sites. The crossover procedure is illustrated in figure 4 for a single crossing site where there are 12 bits making up the parents, and the crossing site was chosen to cut the string between the eighth and ninth bits.

For the problem at hand, the number of crossover sites is initially relatively large, and is decreased in steps when the current number of generations equals multiples of one quarter of the maximum number of generations. This allows the algorithm to form short bit sequences in early generations, while subsequently allowing longer bit strings to form with a higher probability of surviving to subsequent generations.

Once all crossover operations are completed, several options are available for handling the disposition of the children. For instance, they can be accumulated until an arbitrary number of children are produced, at which time they can either replace the current population; or they can be added immediately to the current population, thus increasing the size of the population. Here, the former method of total population replacement was used; the population size was held constant.

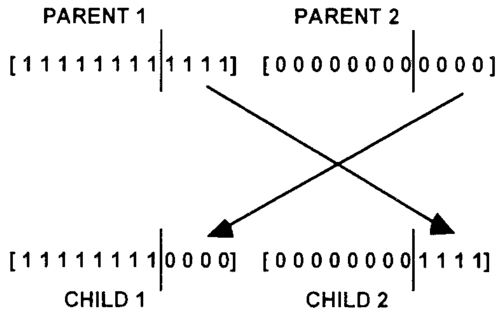


Figure 4. Illustration of Crossover

MUTATION

A mutation operator is included to preserve genetic information; that is, if important genetic information is bred out of the population, a mutation of the genes can reintroduce this information. Mutation is performed randomly as follows. For all samples, starting at the first bit, a uniformly distributed random number is compared to a threshold. If the random number exceeds the threshold, the bit is complemented; i.e., for the i^{th} bit and mutation threshold υ_m,

$$b_i = \overline{b_i} \text{ if } \left(\upsilon_m < U[0,1]\right) . \tag{21}$$

Regardless of the outcome, the same procedure is applied until all bits in the population are visited. The following example illustrates the utility of the mutation operator; if a zero was needed in the third position in the bit string to achieve high performance or minimize the cost, but all bit sequences contained a one in this position, no combination of the parent selection or crossover operators would be sufficient to solve the problem. Therefore, a mutation would be required to complement this bit. An illustration of this mutation is shown in figure 5.

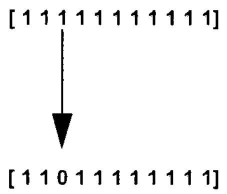


Figure 5. Illustration of Mutation

A high probability of mutation effectively destroys the bit sequences, yielding an inefficient search mechanism. For this reason, to allow the bit strings to stabilize, the probability of mutation is typically less than 10 percent and is adjusted in a manner similar to changing the number of crossover sites.

GENETIC ALGORITHM FOR TARGET TRACKING

In applying GAs to the target tracking problem, a binary representation of the state parameters is used, and the search procedure to minimize the cost is performed iteratively. Each member of the population corresponds to a sample of the state space. Let k_j represent the number of bits for the j^{th} component of the contact state vector, defined in equation (1a). Thus, for the i^{th} sample,

$$\begin{matrix} Rx_{Tbi} \\ Ry_{Tbi} \\ Vx_{Tbi} \\ Vy_{Tbi} \end{matrix}\Bigg] = \begin{bmatrix} b_{o\,Rxi} & b_{1\,Rxi} & \cdots & b_{k\,Rxi} \\ b_{o\,Ryi} & b_{1\,Ryi} & \cdots & b_{k\,Ryi} \\ b_{oVxi} & b_{1Vxi} & \cdots & b_{kVxi} \\ b_{oVyi} & b_{1Vyi} & \cdots & b_{kVyi} \end{bmatrix}, \tag{22}$$

where an unsigned coding scheme was chosen. Here, b_{o_j} is the most significant bit, b_{k_j} is the least significant bit, and the sign information is taken care of when converting to real numbers. The binary representation of the target state is then constructed by concatenating the binary representation of the state variables into a single binary sequence as

$$X_{Tbi} = \left[Rx_{Tbi}\ Ry_{Tbi}\ Vx_{Tbi}\ Vy_{Tbi} \right] . \tag{23}$$

The algorithm is first initialized by randomly distributing ones and zeros in the binary target state strings, X_{Tbi}, for all samples in the population, where the probability of any bit being a one is 50 percent and is independent of the other bits in the population. Let the performance function be defined as in equations (15) through (19).

The parent selection, crossover and mutation operations are subsequently applied in an iterative manner to find the maxima in the performance function, which is equivalent to solving equation (7). For each iteration or generation, the performance is computed for each sample. Parent selection and crossover are next performed $P\!\!\diagup_2$ times. This generates a new population of P samples which replace the parent population. Mutation is performed and the performance of those few samples that were mutated is computed. This process continues until stopping criteria are met. These criteria can include: a maximum number of generations is reached, a maximum performance value (minimum cost) is reached, or the population is determined to have stabilized. The GA target tracking algorithm is summarized in table 2.

Table 2. Summary of Genetic Algorithm

1: Generate P randomly distributed samples of the target state space.	2: Compute the cost for all samples in the population.
3: Compare the cost of all samples to a threshold; if the cost of any one is less than the threshold, stop; otherwise, go to step 4.	4: Compute the weighted and normalized performance values for all samples.
5: Select parents for crossover.	6: Perform crossover for the pairs produced in step 5.
7 Compute the cost of all the samples in the new generation.	8: Compare the cost of the new samples to a threshold; if the cost of any one is below the threshold, stop; otherwise, go to step 9.
9: Compute the weighted and normalized performance for the new samples.	10: Replace original population.
11: Perform mutation.	12: Go to step 2.

The estimation process employing genetic algorithms is illustrated in figure 6.

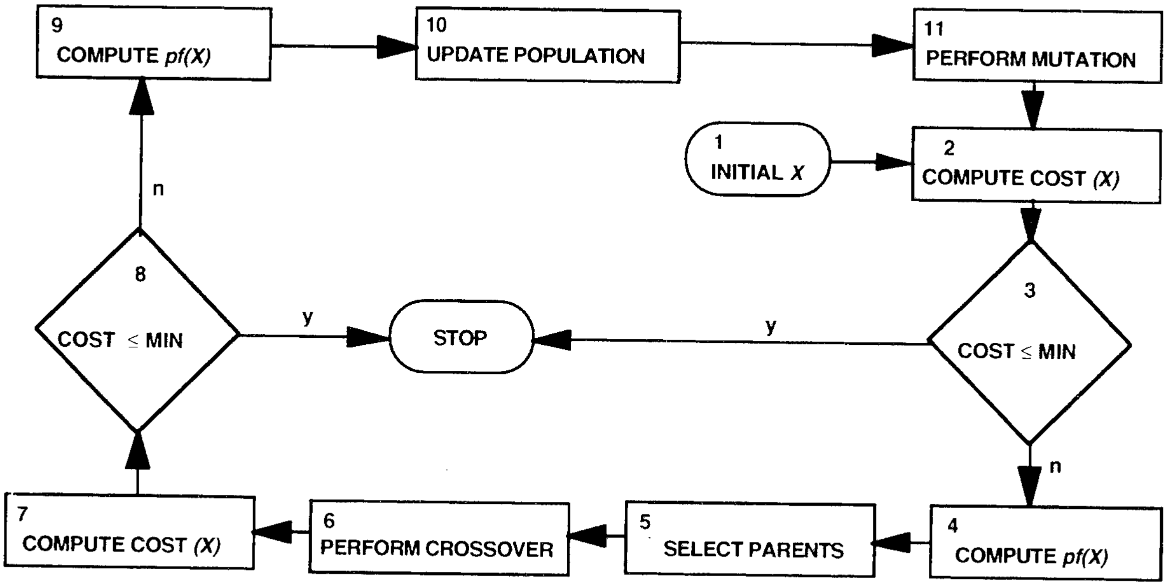


Figure 6. Genetic Algorithm Flowchart

Note that while the flow diagram shows the cost computed twice for the population in every generation, only a few samples get changed by mutation, so the cost only needs to be computed for those few.

While the application of GA to estimate target state parameters is similar to a grid-based search in that it searches from a sampling of the target state space, it has been theoretically determined that the GA searches an equivalent of n^3 data points (reference 15), where n is the total number of points in the state space the algorithm visits. This is because the algorithm concentrates its search more in the areas of maxima that it finds; i.e., it is as if the resolution of a nonuniform grid is dependent upon the value of the density function at the grid points.

To facilitate computational efficiency in subsequent stages of the target tracking system, the weighted centroids of K clusters are computed from the P final solutions, with k defined as the desired maximum number of clusters. The centroids are computed using the performance values as weights. The centroid of the q^{th} cluster is computed by first determining the Euclidean distance between all samples as

$$D_c\left(\hat{X}_i, \hat{X}_j\right) = \left\|\hat{X}_i - \hat{X}_j\right\|^2 \qquad i = 1,2,\ldots,P \qquad j = 1,2,\ldots,P \qquad j \neq i\ . \tag{24}$$

Starting at the closest pair, i.e., $\min\left[D_c\left(\hat{X}_i, \hat{X}_j\right)\right]$, the distance of the sample and the next closest distance to the current cluster is compared. If the distance to the next sample is greater than a specified percentage of the previous distance, start a new cluster; otherwise add this sample to the cluster and search for the sample with the next closest distance to the current cluster. This procedure is iterated until all samples are assigned to clusters. If the number of clusters at any point exceeds K, the specified percent change in the distance allowed between samples within a cluster is relaxed and the algorithm starts again. Once the clusters are formed, the centroids of all clusters are computed as follows. The centroid for the q^{th} cluster, $\hat{X}_q$, is computed as

$$\hat{X}_q = \frac{1.0}{N_q}\sum_{i=1}^{N_q}\hat{X}_i perf\left(\hat{X}_i\right), \tag{25}$$

where N_q is the number of samples in the cluster.

VI. SIMULATION EXPERIMENTS

To illustrate the potential performance of SA and GA as tracking algorithms, experiments corresponding to both good and adverse conditions of surface ship target tracking using simulated active range and bearing measurements are conducted. For comparison purposes MLE results are also presented. For these experiments, knowledge of the environmental and sensor models are assumed, and the data are assumed to have been properly associated and are corrupted by zero mean Gaussian white noise with known variance.

Results are presented as polar scatter plots in both range-bearing and speed-course state spaces. The average error and standard deviation of the error in the estimates are computed for the MLE and SA estimators. Because the GA estimator returns multiple estimates, the cumulative performance values are also plotted as histograms for each of the polar coordinate states.

EXPERIMENTAL DESCRIPTION

The surface ship geometry used for both experiments is depicted in figure 7 and summarized in table 3. The observer starts at the origin and has heading and speed of 26° and 12 knots, respectively. It maintains a constant speed and traverses two 20-minute legs with an instantaneous course maneuver to 154° at 20 minutes. The target has a constant course of 270° and speed of 8 knots.

Table 3. Geometry Description

Geometry	Time on Leg (min)	Contact Course (deg)	Contact Speed (knots)	Observer Course (deg)	Observer Speed (knots)	Initial Bearing (deg)	Initial Range (km)
80-sample	20 20	270	8	26 154	12	79	20
10-sample	20 2.5	270	8	26 154	12	79	20

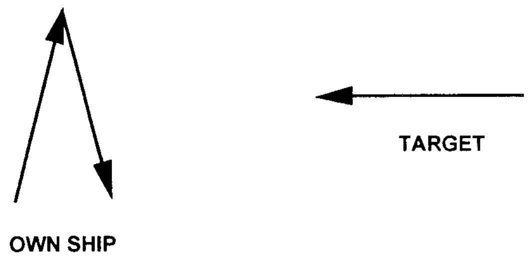


Figure 7. Experimental Geometry

Monte Carlo simulations were conducted with 100 noise sequences for cases involving 80-sample and 10-sample data sets with a 32-second sampling period. Active bearing measurements were simulated with noise variances of 4.0 deg^2 and 1000 km^2, respectively, for the 80-sample scenario, and noise variance of 4.0 deg^2 and 9x10^7 km^2 for the 10-sample geometry. The latter case essentially represents a bearings-only scenario. As such the 80-sample geometry represents a scenario with good observability properties, while the 10-sample geometry represents a scenario with poor observability. It is noted that measurements are only available for the first part of each leg of the 10-sample geometry.

The SA and GA parameters used for these experiments are shown in tables 4 and 5 respectively. The MLE and SA estimators are initialized with the measured bearing and range.

Table 4. Simulated Annealing Parameter Values

I_t	10
α	0.94
T_o	100
T_{min}	10^{-6}
σ^2_{Rx}	10^4 m^2
σ^2_{Ry}	10^4 m^2
σ^2_{Vx}	0.01 m^2/sec^2
σ^2_{Vy}	0.01 m^2/sec^2
No. of Temperatures	352

Table 5. Genetic Algorithm Parameter Values

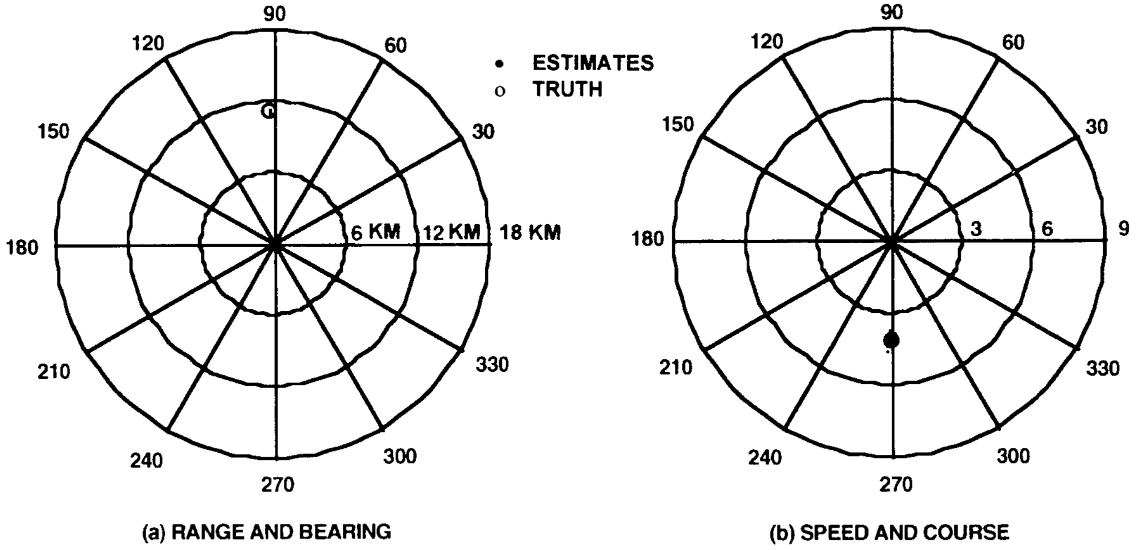
Rx_{max}	30 km	Rx_{min}	-30 km	no. of bits	12
Ry_{max}	30 km	Ry_{min}	-30 km	no. of bits	12
Vx_{max}	40 m/sec	Vx_{min}	-40 m/sec	no. of bits	8
Vy_{max}	40 m/sec	Vy_{min}	-40 m/sec	no. of bits	8
Population size	34				
Max no. of generations	352				
Probability of crossover	99%				
No. of crossing sites	initial	4	final	2	
Probability of mutation	initial	3%	final	0.1%	

RESULTS AND DISCUSSION

Results for the 80-sample data set case are illustrated in figures 8 through 12 and tables 6 and 7. This represents a reasonably favorable tracking condition. As such, the MLE results shown in figure 8 and table 6 indicate that consistent and accurate contact solutions are achieved. As can be seen from figures 9 through 12 and table 7, similar performance is realized by both SA and GA tracking algorithms. It is noted that while the GA estimator appears to have a larger scatter than the MLE or SA estimators, examination of figures 11 and 12 indicates the scatter is very tight.

Table 6. MLE Error Statistics for the 80-Sample Data Set Case

	Range (km)	Bearing (deg)	Course (deg)	Speed (m/sec)
Average Error	-0.008	-0.07	-0.08	-0.04
Standard Deviation	0.1094	0.24	1.15	0.15



(a) RANGE AND BEARING

(b) SPEED AND COURSE

Figure 8. MLE Estimates for the 80-Sample Data Set Case

Table 7. SA Error Statistics for the 80-Sample Data Set Case

	Range (km)	Bearing (deg)	Course (deg)	Speed (m/sec)
Average Error	-4.156	-8.26	-0.88	-0.16
Standard Deviation	0.1501	1.60	1.40	0.45

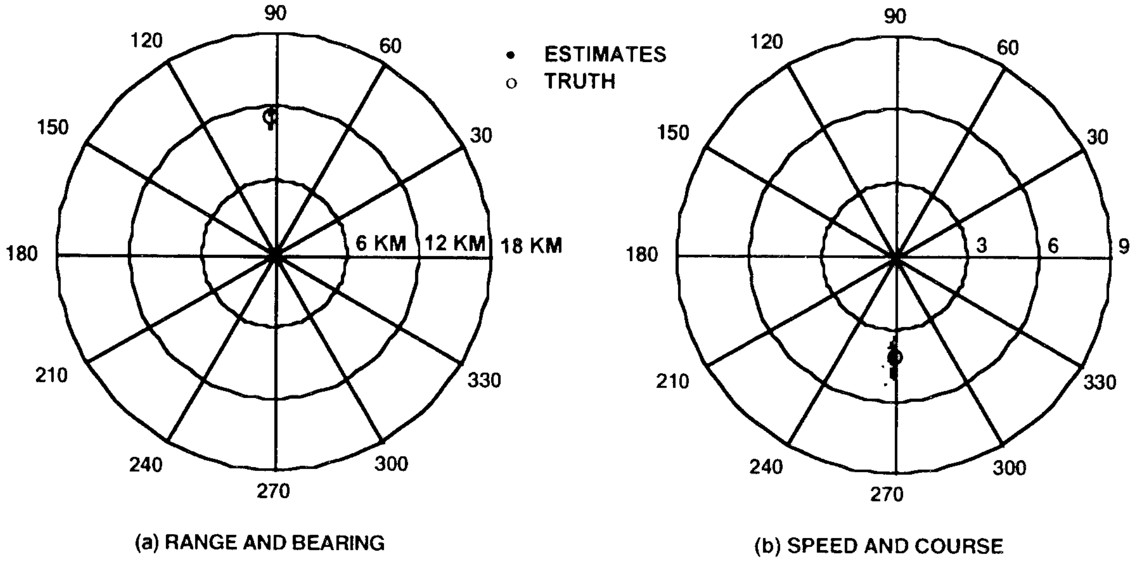


Figure 9. SA Estimates for the 80-Sample Data Set Case

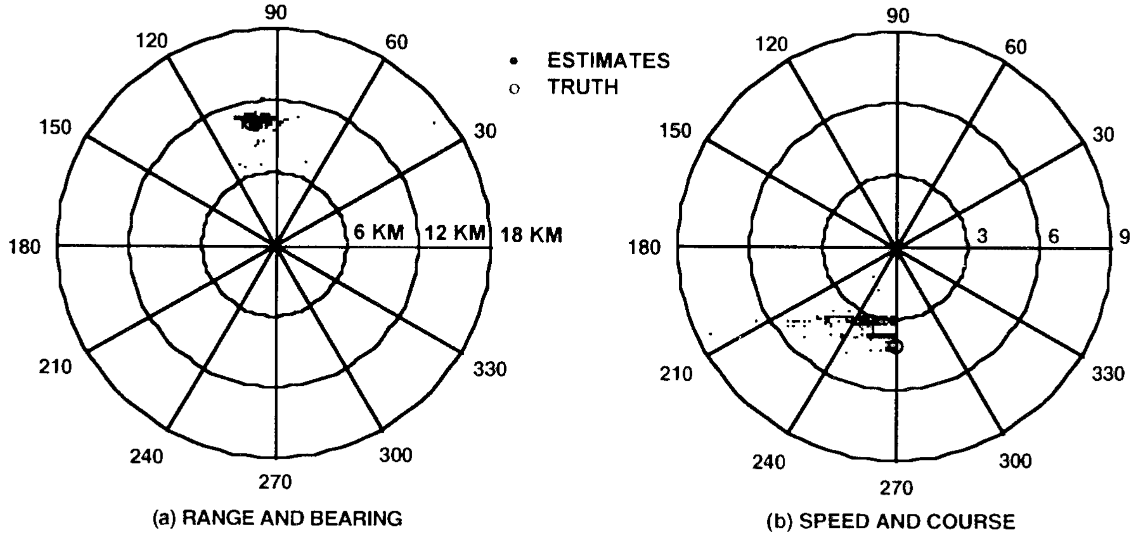


Figure 10. GA Estimates for the 80-Sample Data Set Case

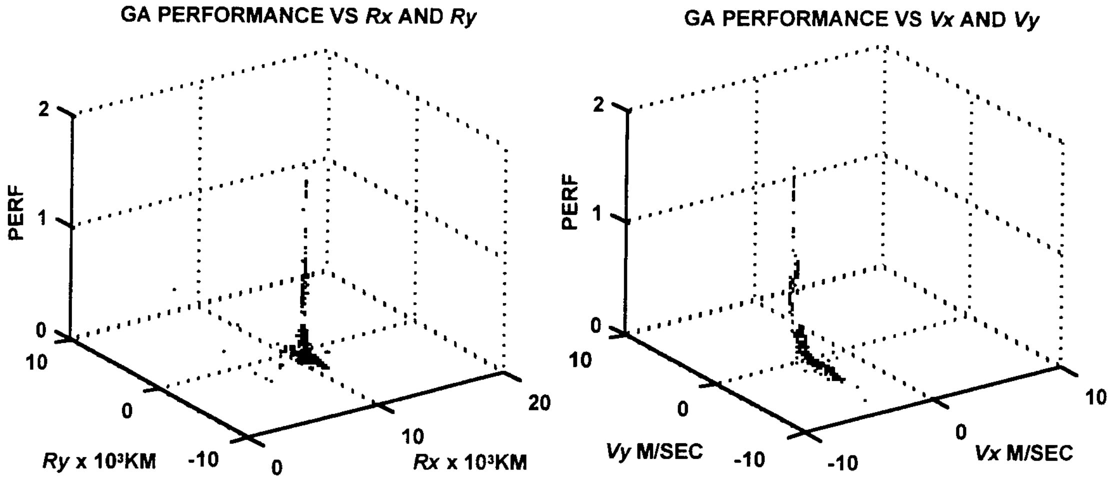


Figure 11. GA Performance Density for the 80-Sample Data Set Case

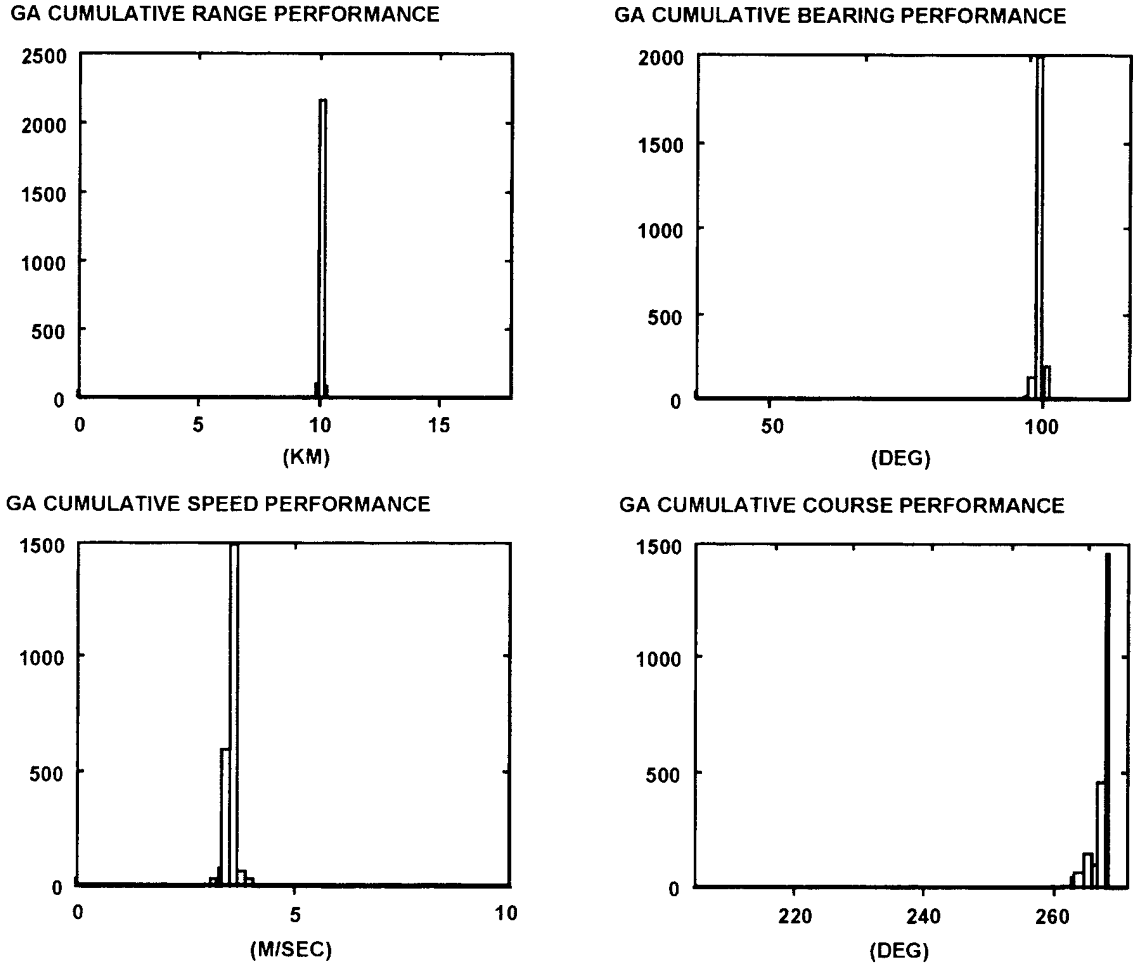
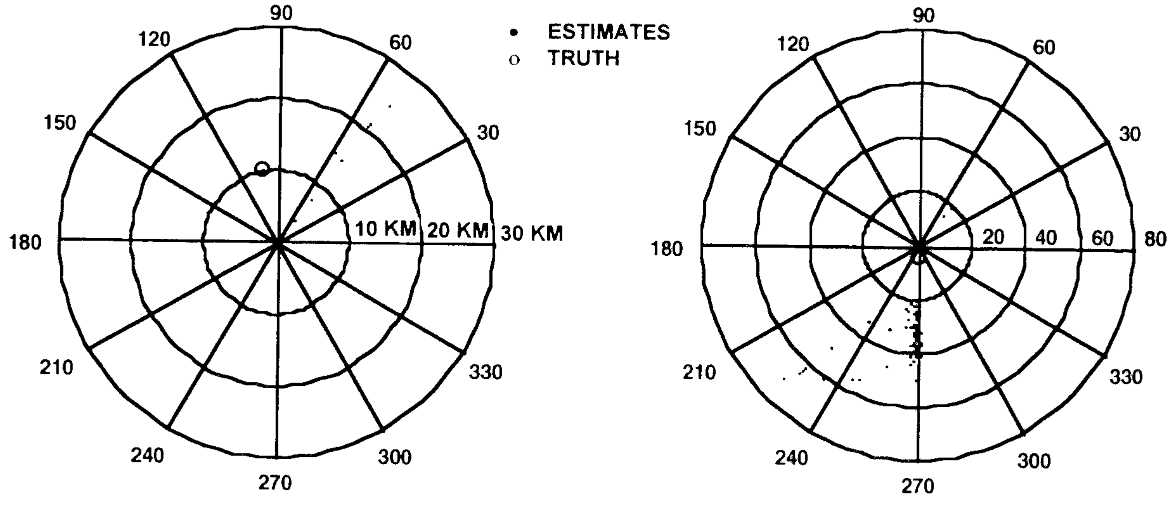


Figure 12. GA Cumulative Performance Distributions for the 80-Sample Data Set Case

Results for the low observability case are shown in figures 13 through 17 and tables 8 and 9. The collapse of the estimates to the origin in figure 13, and the large errors exhibited in table 6 are evidence that the MLE has a tendency to diverge under high noise, sparse and intermittent conditions. On the other hand, figures 14 through 17 and table 9 show that SA and GA estimates are well behaved and are clustered about the truth.

Table 8. MLE Error Statistics for the 10-Sample Data Set Case (Poorly Observable)

	Range (km)	Bearing (deg)	Course (deg)	Speed (m/sec)
Average Error	-9.127	-59.3	-11.9	30.8
Standard Deviation	4.207	33.5	39.4	10.1



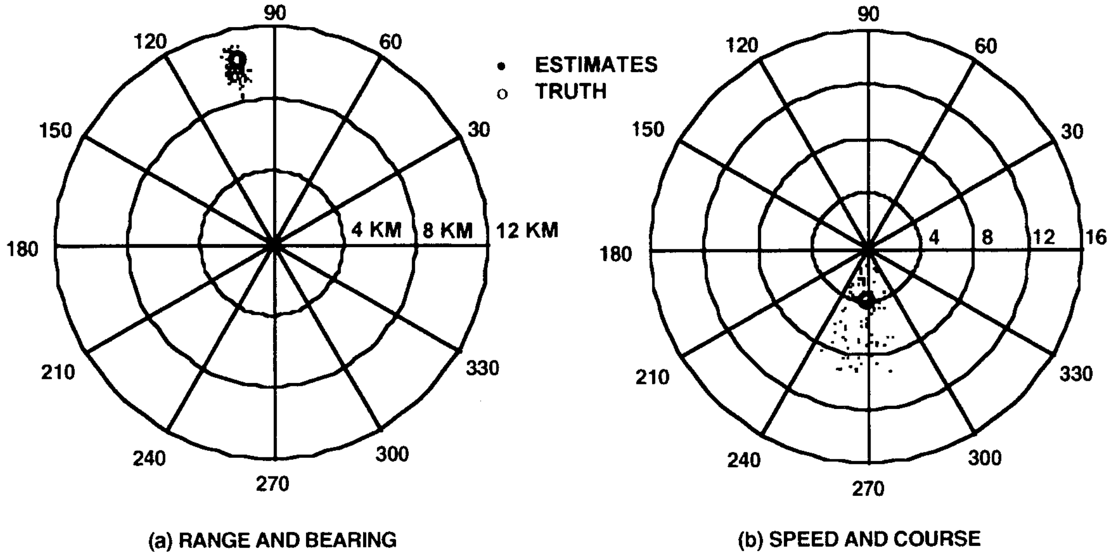
(a) RANGE AND BEARING

(b) SPEED AND COURSE

Figure 13. MLE Estimates for the 10-Sample Data Set Case

Table 9. SA Error Statistics for the 10-Sample Data Set Case (Poorly Observable)

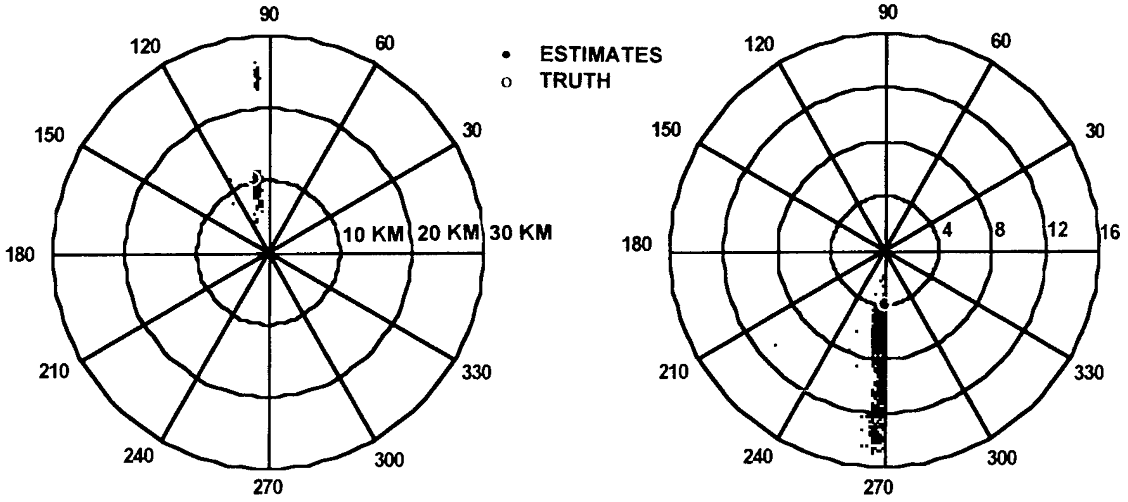
	Range (km)	Bearing (deg)	Course (deg)	Speed (m/sec)
Average Error	-0.145	1.49	-11.23	0.34
Standard Deviation	0.664	1.9	36.9	2.8



(a) RANGE AND BEARING

(b) SPEED AND COURSE

Figure 14. SA Estimates for the 10-Sample Data Set Case



(a) RANGE AND BEARING

(b) SPEED AND COURSE

Figure 15. GA Estimates for the 10-Sample Data Set Case

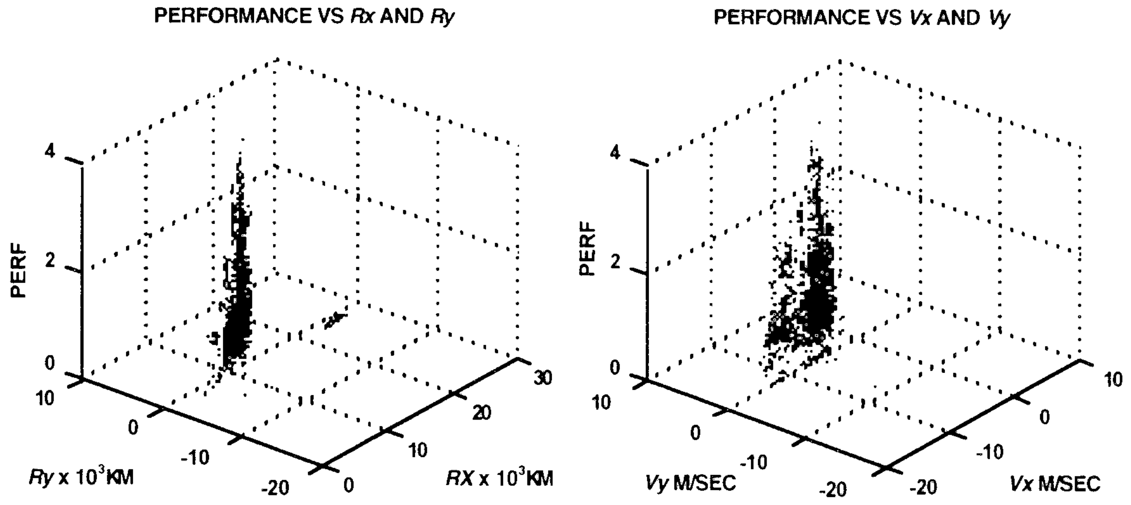


Figure 16. GA Performance Densities for the 10-Sample Data Set Case (Poorly Observable)

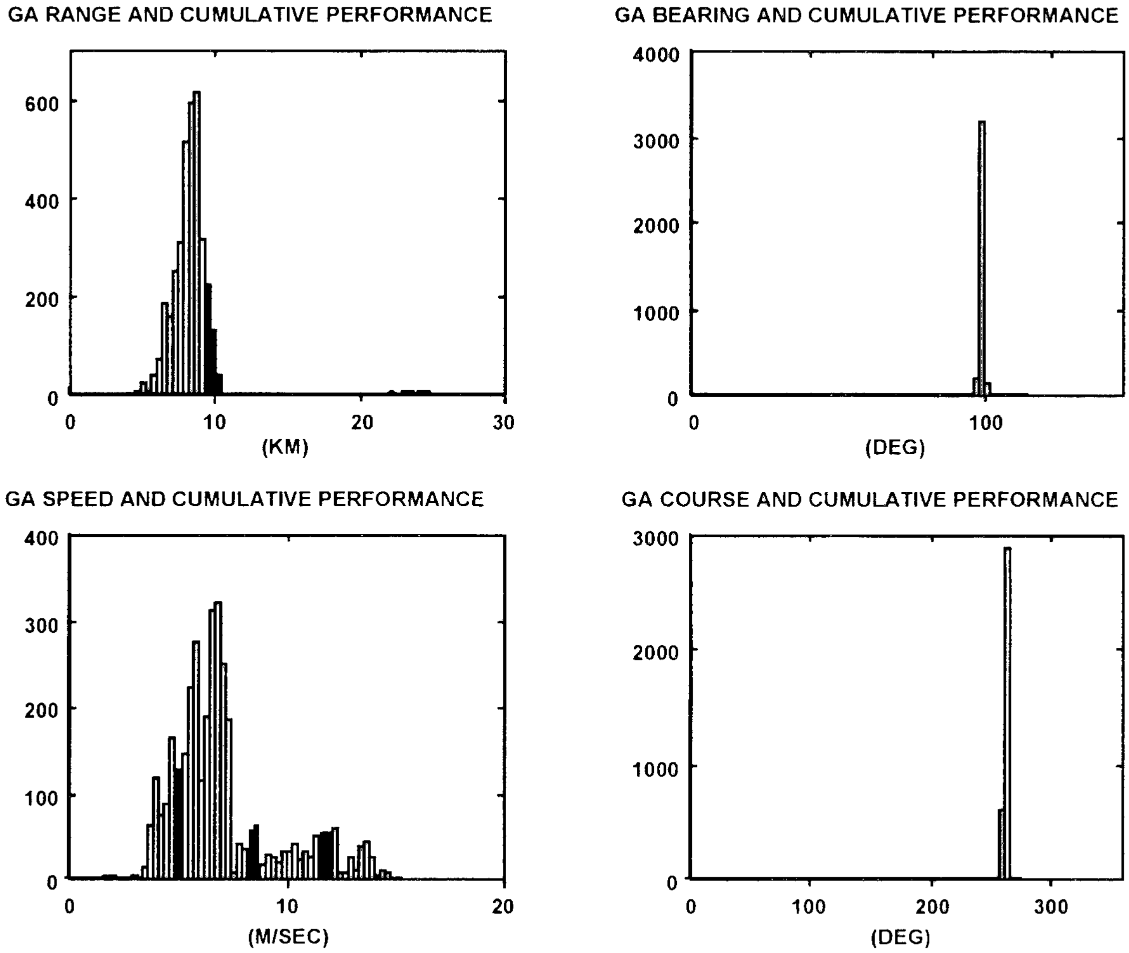


Figure 17. GA Cumulative Performance Distribution for the 10-Sample Data Set Case (Poorly Observable)

To examine the potential computational efficiency of SA and GA, we compare the estimated number of times each algorithm evaluates the performance function to achieve the desired convergence with the conventional grid-search technique. With a maximum of 2 re-annealings, 10 iterations per temperature update, and 352 temperature changes for the SA estimator, and with a population size of 34 for 352 populations for the GA estimator, the number of performance evaluations is approximately 10,000 and 12,000, respectively. For the standard grid-search technique, if we partition the states with a 30x30 position and 10x10 velocity grid, and apply 3 passes, with a finer resolution of the same number of grid points for each pass, the number of performance evaluations would be 270,000, or an order of magnitude more computations than SA and GA. It is noted that this is a conservative evaluation, and the gain in processing efficiency can be reduced further by optimizing the search parameters of the SA and GA algorithms and by applying a more robust stopping criteria.

VII. SUMMARY AND CONCLUSIONS

The applicability of random search techniques for contact tracking, including SA and GA, has been examined. The potential of these techniques, in terms of efficacy and efficiency, when applied as either stand-alone or in conjunction with other conventional systems, was analyzed. A comparison was also made between the behavior of the SA and GA estimators and the traditional gradient-based and grid-based algorithms under both good and poor observability conditions. The behavior of these algorithms was demonstrated through simulation experiments involving surface ship tracking with active data.

Simulation results were presented for two cases; one scenario which was data rich and had good observability properties, and a second which had sparse data and poor observability properties. The MLE, SA and GA estimators had similar performance under good observability conditions. However, for the poorly observable case, the MLE diverged while SA and GA provided good estimates clustered around the vicinity of the truth. While computational efficiency of the SA and GA estimators has not been optimized, a conservative analysis showed both GA and SA to be significantly more efficient than the traditional grid-based algorithm.

SA warrants further study as a contact state estimation technique. One drawback in the current implementation occurs when multiple maxima are present in the state density and the value of the density at the maxima are approximately equivalent; it will only find one solution , with no indication of other maxima in the final result. This can be a cause for concern, since only one of the local maxima contains the truth, thus they may all need to be found. Also, while simulated annealing provides a mechanism for searching a state density without getting trapped in local minima, its performance in this aspect is highly dependent on the temperature schedule. This aspect will require further study.

As has been discussed earlier, genetic algorithms can provide an indication of the location and relative importance of multiple maxima in a global density function (reference 15), however, a sufficient number of samples in the population must be employed in order to determine not only the number of maxima, but also how fast the algorithm converges.

Thus, while applying SA or GA to contact state estimation appears promising, and can potentially alleviate some of the problems associated with traditional estimation algorithms, more analysis needs to be conducted to optimize the parameters which control the behavior of these algorithms and to determine the best combination of algorithms for contact tracking applications.

REFERENCES

1. D. Kolb and F. Holister, "Bearings-only Target Parameter Estimation," *Proceedings of the First Asilomar Conference on Circuits and Systems*, 1967.

2. K.F. Gong et al., "Three Dimensional Target Parameter Estimation," *Proceedings of the Sixteenth Asilomar Conference on Circuits, Systems and Computers*, 1982.

3. S.C. Nardone and V. J. Aidala, "Observability Criteria for Bearings-only Target Motion Analysis," *IEEE Transaction on AES*, vol. AES-17, November 1980.

4. M.L. Graham, K.F. Gong, N.A. Jackson, and J.G. Baylog, "Lower Bound Analysis for Large Errors in Nonlinear State Estimation," *Proceedings of the 22nd Asilomar Conference on Signals, Systems and Computers,* November 1988.

5. S.S. Blackman, *Multiple Target Tracking With Radar Applications*, Artech House, Norwood, MA, 1986.

6. Y. Bar-Shalom and T.E. Fortman, *Tracking and Data Association*, Academic Press, Orlando, FL, 1988.

7. H. Sorenson, *Parameter Estimation Principles and Problems*, Vol. 9, Marcel Dekker, New York, 1980.

8. A.H. Jazwinski, *Stochastic Process and Filtering Theory*, Academic Press, New York, 1970.

9. H.L. Van Trees, *Detection Estimation and Modulation Theory*, Part I, John Wiley and Sons, New York, 1968.

10. V.J. Aidala, "Kalman Filter Behavior in Bearings-Only Tracking Applications," *IEEE Transaction on Aerospace and Electronic Systems*, vol. AES-15, no. 1, January 1979.

11. S.E. Hammel et al., "Recursive vs. Batch Processing Algorithms for Bearings-Only Tracking," *Proceedings of the 1983 International Ocean Engineering Conference*, August 1983.

12. J.G. Baylog et al., "Underwater Tracking in the Presence of Modeling Uncertainty," *Proceedings of the 21st Asilomar Conference on Signals, Systems and Computer*s, November 1987.

13. S. Kirkpatrick, C. Gelatt Jr., and M. Vechi, "Optimization by Simulated Annealing," *Science*, vol. 220, no. 4598, 13 May 1983.

14. M. Wilhelm and T. Ward, "Solving Quadratic Assignment Problems by Simulated Annealing," *IIE Transactions*, March 1987.

15. D. Goldberg, *Genetic Algorithms in Search Optimization and Machine Learning,* Addison-Wesley, Reading, MA, 1989.

INITIAL DISTRIBUTION LIST

Addressee	No. of Copies
Defense Technical Information Center (DTIC)	12
Chief of Naval Research (ONR-333, J. Fein; ONR-342, T. McMullen)	2
Center for Naval Analyses (CNA)	1
Naval Research Laboratory (NRL)	1
Naval Postgraduate School (NPS)	1
Naval War College (NWC)	1
Naval Sea Systems Command (PEO-USW, PEO-SUB, PEO-SUB-R, PEO-SUB-X, ASTO-E, PMS-350, PMS-393, PMO-403, PMO-418)	9
NSWC White Oak (N51, M. Williams)	1